

Progettini per il corso IR 2012.

Per domande inviare email sia al sottoscritto sia a vitaled@gmail.com

Progetto 1.

L'obiettivo del progetto è quello di sperimentare una serie di algoritmi di clustering su un dataset di news in italiano. Ogni gruppo di lavoro (formato da 2 o 3 persone) esperimenterà un solo algoritmo, concordato con il docente. I file sono descritti qui di seguito, come di seguito è indicato il sito da cui scaricarli.

Dataset: Avrete a disposizione vari file, ciascuno contenente un insieme di news di categoria diversa (mondo, italia, sport, spettacolo, scienza&tecnologia, economia) e un clustering già calcolato su ciascuno di questi file.

Ogni file contiene 5000 news, ed è strutturato per righe, ciascuna avente la forma:

```
[epoch]
[\\t]
[id_sorgente]
[\\t]
[testata giornalistica]
[\\t]
[titolo news]
[\\t]
[corpo news]
[\\t]
[id_cat]
[\\t]
[link notizia]
```

Ogni file di clustering è strutturato per righe, ciascuna avente la forma:

```
[intero non significativo]
[\\t]
[sequenza termini importanti del cluster, divisi da spazi] Possono anche essere inserite più sequenze nel
caso i termini siano divisi in insiemi, in tal caso dividere le sequenze con una virgola ",".
[\\t]
[sequenza ID news ordinate per importanza nel cluster]
[\\t]
[flag binario: 1 cluster valido, 0 cluster non valido]
```

Gli identificativi delle news fanno riferimento al loro ordinamento all'interno del file di news e sono conteggiati a partire da 1.

Il flag binario deve indicare se il cluster è interessante o no. E' auspicabile che i cluster siano ordinati in ordine decrescente di importanza. Se ciò non è possibile ordinarli in modo decrescente di size.

Stessa cosa dicasi per le news di ogni cluster, devono essere ordinate per importanza decrescente, o se non ci riuscite, ordinatele dalla più recente alla meno recente.

Clustering: Si vuole progettare un algoritmo di clustering in C o C++ che eventualmente sfrutta librerie open-source, p.e. GPL, tra quelle indicate qui di seguito. Altre proposte devono essere concordate con il docente.

1. Libreria Gmeans. <http://www.dataminingresearch.com/index.php/2010/06/gmeans-clustering-software-compatible-with-gcc-4/>
2. Co-clustering. <http://www.cs.utexas.edu/users/dml/Software/cocluster.html>
3. Riscrivere in C++ il codice dell'optimal bisect <http://code.google.com/p/airhead-research/source/browse/branches/cluster/src/edu/ucla/sspace/clustering/BisectingKMeans.java?r=1046>.
4. Costruire un grafo di news, con arco che indica la misura di jaccard su shingles di lunghezza 3 per i termini stemmati. Le shingles potrebbero forse essere pesate con una sorta di TF-IDF (da valutare). Poi si fa clustering, ordinando gli archi per peso crescente, ed eliminando questi nell'ordine a partire da quelli di peso minore, e via via a crescere fintanto che si ottengono k clusters o l'arco da eliminare è maggiore di una certa soglia.
5. Latent Semantic Indexing: <http://gibbslda.sourceforge.net/>

Varie osservazioni:

- Si potrebbe pensare di usare come features non solo i singoli termini ma shingles di 3 termini ciascuno, eventualmente stemmati. O solo le shingles, senza i singoli termini.
- Visto che i clusters lo prevedono, potreste studiare se il metodo adottato offre un modo semplice ma efficace per determinare i termini significativi di ogni cluster. Altrimenti lasciate la riga vuota.

Visualizzazione: Fare una valutazione dei propri risultati con quelli forniti a esempio per la bontà della soluzione proposta. Il confronto è “a occhio” e deve essere fatto in modo oggettivo; l'accettazione del progetto non dipende dall'aver trovato un risultato migliore, ma di aver condotto un'analisi completa e seria.

Per la valutazione è stata messa a punto un'interfaccia grafica, mediante browser. Per utilizzarla dovete avere installato sulla vostra macchina un web server apache con il supporto per PHP. Se non lo avete, e non avete particolare dimestichezza con questo tipo di tecnologie, potete scaricare XAMPP, un pacchetto già pronto che include tutto il necessario:

<http://www.apachefriends.org/it/xampp.html> (disponibile Linux/Windows/Mac OS X)

Una volta installato l'ambiente dovete scompattare il file “**cui.tgz**” all'interno nella cartella **www** di apache. L'indirizzo dal quale scaricare il file è: <http://zola.di.unipi.it/acube/repo/cui.tgz>

Aprirete quindi un browser (consigliati firefox o chrome) e collegatevi all'indirizzo:

<http://localhost/cui/index.php> per visualizzare l'interfaccia.

All'interno della cartella **www/cui/** troverete due sottocartelle **dataset** e **cluster**. A prima cartella è popolata da i file delle news su cui dovete fare il clustering, la seconda contiene degli esempi di clustering già fatti e contro i quali vi dovrete confrontare.

Nella cartella **cluster** andranno tutte le soluzioni di clustering da voi proposte. I nomi dei file nella cartella **clusters** devono avere lo stesso nome del file della cartella **dataset** cui il clustering si riferisce e seguiti da **_id**.

Esempio:

dataset/sport

cluster/sport_0

Per ogni file di input in **dataset** potete quindi generare più soluzioni di clustering semplicemente aggiungendo un nuovo file alla cartella **cluster**, e poi visualizzarli singolarmente scorrendo la tendina.

L'interfaccia vi permetterà di scegliere la soluzione che volete visualizzare, e le news dentro il clustering selezionato verranno ordinate secondo il vostro ranking (ordine dato nel file del cluster, es. sport_0), mostrandovi per ogni cluster le 3 news più importanti. Questo vi darà già un'idea di massima della qualità della vostra soluzione, potrete poi estendere la visualizzazione per ogni singolo cluster per valutare tutte le news cliccando su pulsante alla fine di ogni cluster.

All'interno della cartella **cui** troverete anche una cartella **stemmed** che contiene dei file con la stessa struttura di quelli riportati nella cartella **dataset** ma nei quali titolo e descrizione della news sono stati sottoposti a stemming (riduzione di una parola alla sua radice linguistica).

Potete usare sia la versione in **dataset** che in **stemmed**, a seconda che la vostra strategia preveda o no una fase di stemming. Potete quindi anche valutare se sfruttare una qualche proprietà dei documenti che andrebbe persa in seguito allo stemming dei termini.

Progetto 2.

I dataset sono i clusters forniti nel punto precedente.

L'obiettivo del progetto è quello di calcolare clique di termini di size 5 di ogni cluster di news, con una valutazione della bontà di ogni clique trovata (possono essere anche più di una per cluster). Una clique è definita come: "Represents the most descriptive features via a graph in which two features are connected via an edge if and only if their co-occurrence frequency within the cluster is greater than their expected co-occurrence." Alcune librerie per determinare clique su grafi si trovano su:

http://en.wikipedia.org/wiki/Clique_problem, dove è anche indicato un semplice algoritmo per calcolare la clique di dimensione 3 (triangoli: Cliques of fixed size).

I triangoli poi possono essere estesi a clique di dimensione 4 e poi 5. Interessante è anche la libreria <http://igraph.sourceforge.net/> per il calcolo delle clique oppure per altre strutture che esprimono coesione tra termini (community?).

Progetto 3.

Progettare un crawler (in perl?) che scarica a intervalli di tempo predefiniti le rassegne stampa di quotidiani nazionali. Ad esempio:

<http://rstampapubblica.istruzione.it/rassegna/rassegna.asp>

<http://rsegnastampa.unipi.it/sup/>

http://www.interno.gov.it/mininterno/site/it/sezioni/sala_stampapubblica/rassegna_stampapubblica/

(sarebbe bene che voi trovaste anche altre sorgenti di valore da cui scaricare altre rassegne)

L'idea è quella di costruire un db con un record per rassegna stampa, quindi contenente la data, la testata giornalistica, il titolo, e il link al pdf (o anche il pdf stesso, scaricato in locale), eventualmente la categoria della news (top, politica, sport, spettacolo,...) se individuata nella rassegna medesima.

Un compito è quello di trovare più siti di rassegne e quindi scaricare più cose, eventualmente eliminando i duplicati.

Come extra potreste sviluppare un semplicissimo algoritmo che cerca il match tra una rassegna e una news tra quelle disponibili nei file in **dataset**, identificando quella più pertinente, p.e. basandosi sull'uguaglianza della sorgente, sulla quasi-uguaglianza del titolo [può accadere che una news dal feed rss al giornale stampato cambi il titolo, ciò però deve essere un piccolissimo cambio per essere accettabile].

Il test potrebbe consistere nel ricevere in input un file di news, e per ciascuna cercare nel DB-rassegna quella che corrisponde alla news, se c'è, altrimenti non restituire nulla. Nel caso di match si stampa a video la news e il link al pdf, tanto per poter fare verifiche a occhio.